Knowledge-based expert system using a set of rules to assist a tele-operated mobile robot

Sanders, David Adrian; Gegov, Alexander; Ndzi, David

Published in:
Intelligent Systems and Applications

DOI:
10.1007/978-3-319-69266-1_18

E-pub ahead of print: 31/12/2017

Document Version
Peer reviewed version

Citation for published version (APA):
Sanders, D. A., Gegov, A., & Ndzi, D. (2017). Knowledge-based expert system using a set of rules to assist a tele-operated mobile robot. In Y. Bi, S. Kapoor, & R. Bhatia (Eds.), *Intelligent Systems and Applications: Proceedings of SAI Intelligent Systems Conference* (Vol. 751, pp. 371-392). (Studies in Computational Intelligence; Vol. 751). Springer. https://doi.org/10.1007/978-3-319-69266-1_18

Knowledge-based expert system using a set of rules to assist a tele-operated mobile robot

David Adrian Sanders[1], Alexander Gegov[1] and David Ndzi[2]

1. Faculty of Technology, Portsmouth University

2. School of Engineering and Computing, University of the West of Scotland

tele-op-research@serg.org.uk

Abstract — This paper firstly reviews five artificial intelligence tools that might be useful in helping tele-operators to drive mobile robots: knowledge-based systems (including rule based systems and case-based reasoning), automatic knowledge acquisition, fuzzy logic, neural networks and genetic algorithms. Rule-based systems were selected to provide real time support to tele-operators with their steering because the systems allow tele-operators to be included in the driving as much as possible and to reach their target destination, while helping when needed to avoid an obstacle. A bearing to an end-point is added as an input with an obstacle avoidance sensor system and the usual inputs from a joystick. A recommended direction is combined with the angle and position of a joystick and the rule-based scheme generates a recommended angle to rotate the mobile robot. That recommended angle is then blended with the user input to assist tele-operators with steering their robots in the direction of their destinations.

1. Introduction

Five artificial intelligence tools are reviewed: knowledge-based systems (including rule based systems and case-based reasoning), fuzzy logic, automatic knowledge acquisition, neural networks and genetic algorithms. Each artificial intelligence tool is outlined and briefly reviewed. A Knowledge-based expert system using a set of rules is selected to help tele-operators to drive their mobile robots.

Applications of these tools have become more widespread and more complex mobile robot applications may require greater use of hybrid tools that combine the strengths of two or more of the tools. The tools and methods have minimal computation complexity and can be implemented on single robots or systems with low-capability microcontrollers. The appropriate deployment of the new AI tools will contribute to the creation of more efficient and effective mobile robot and tele-operated systems.

A rule-based system that describes knowledge in terms of IF…THEN…ELSE is selected. The moving machine obtains information about its local surrounding environment from sensors while moving towards a more global destination. Assistance is made available to help tele-operators avoid obstructions.

Systems presented in this paper help tele-operators drive when they cannot see (possibly because of smoke or because the view is obscured) or when the robot is in a remote setting away from the driver.

Tele-operated structures are often open-loop. Operatives communicate their desired direction and speed using a joystick. The robot then tends to move in the desired speed and direction. Tele-operator demands are processed and blended with inputs from the ultrasonics along with a more global destination end point to help the operatives to drive their robots. Local and global planning are mixed inside a knowledge-based expert system using a set of rules to help the operatives to steer their robots. Local information from ultrasonics [1] is blended with a global path.

Navigation for tele-operated mobile robots is discussed within academic literature [1-4]. Usually they have used a local algorithm and aimed to circumvent obstructions [5] and suggest movements based on local sensors[4].

Some work has planned initial paths for mobile robots and then modified them locally [1]. In this work, a local planner produces drive to motors attached to the driving wheels depending on input received from: transducers using ultrasonics, the joystick, and the global targets. The robots respond swiftly to the desires of the operatives and to unanticipated obstructions but has a tendency to move towards the goal objective on every occasion.

Huq defined a fuzzy blending of schemas that depended on context [6] and that eliminated a few restrictions that had become apparent in previous approaches. It used navigation based on goal orientation as well as avoiding obstacles within the robot path. Fuzzy logic has been blended with Genetic algorithms to solve mapping and location problems[7]. That automatically looked for a suitable local plan. Bennewitz & Burgard described a method to create random real time routes within undefined environments without using vision [1], [8], while tracking trajectories [9]. Hwang & Chang described techniques to avoid obstacles that used a fuzzy decentralized sliding-mode of control[10]. The potential field method was improved by Song and Chen by resolving some of the local minima problems [5] and Nguyen defined an obstacle avoidance system that used Bayesian Neural Networks[11].

A technique that improves a minimum-cost route is presented in this paper. A joystick mainly controls the speed but some simple AI systems also provides input [12-15]. The AI methods use rules that were perception based and similar to those described by Parhi and Singh, who used them for an independent self-directed mobile robot [1] and by Sanders *et al* [16] who considered a tele-operated mobile robot.

Algorithms trade path length against distance to an obstacle(s). Rules generate a suggested steering angle and that steering angle is merged with the contribution provided by a joystick to create the signals to drive the mobile robot motors.

The system and the techniques were successfully proven using simulation and then the sensors and microcontrollers were mounted on a mobile robot (Fig. 1).

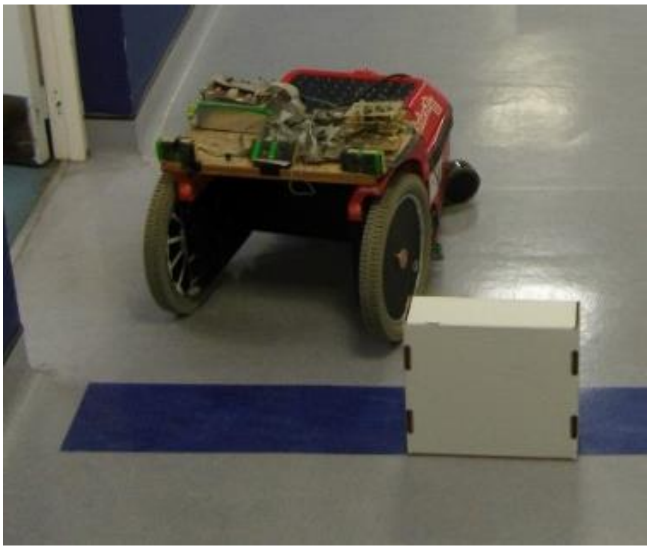

Fig. 1 Bobcat II mobile robot base avoiding an obstacle while being driven along a corridor

Many sensors can be used for obstacle avoidance, for example: structured light or laser [17]; ultrasonics [18]; or infra-red [19]. The more comprehensive methods sometimes perform poorly indoors [20] but simpler and more local sensors can successfully determine position, for example: gyros, odometers, tilt, and ultrasonic [21] [22]. Images of the space ahead of the robot can be converted into a digital format and can be useful when the view ahead of the robot is unobstructed but vision systems can need more processing and they can be more complicated [23]. They are getting cheaper and computing power is quickly increasing [24]. The most accurate source of knowledge about the surroundings and situation comes from the human tele-operator but diminished visibility, separation and imperfect environmental information can reduce the ability of a human teleoperator [25].

Ultrasonics were selected for detecting ranges because it was inexpensive, uncomplicated, straightforward and rugged [26].

The paper continues with a review of the five artificial intelligence tools that were considered for this work followed by a description of the input from the sensors and joystick. Then the kinematics of the mobile robot base are described before discussing control and the artificial intelligence rule based tool selected. Then the testing and the results are described and the paper finishes with some discussion and conclusions.

2. Review of some artificial intelligence tools

Artificial Intelligence (AI) can improve teleoperation of mobile robots. AI has produced some useful tools for teleoperation that automatically solve problems normally requiring human brainpower. Five such tools are reviewed in this Section: fuzzy logic, knowledge-based systems, inductive learning, neural networks and genetic algorithms.

New advances are allowing seamless interactions between computers and people and the introduction of AI into teleoperation promises to make it more flexible, efficient and reliable. Tele operated mobile robots are exceeding human performance and as they merge with humans more intimately and we combine computer capacity with brain power to analyse, deliberate and make decisions, then we might be on the verge of a new assistive robot age.

A. Knowledge-based systems

Knowledge-based systems (sometimes called expert systems) are computer programs representing knowledge about solving problems. These systems typically have two principal parts, knowledge-bases and inference-mechanisms. Knowledge-bases hold knowledge about a domain that can be stated as arrangements of 'IF–THEN' rules, frames, factual statements, procedures, objects and cases.

Inference mechanisms manipulate stored knowledge to generate solutions. Knowledge manipulation methods include using constraints and inheritance (in object-oriented expert systems a frame-based expert systems), recovery and reworking of case examples (in a case-based system) and applying inference rules (within a rule-based system), corresponding to control procedures (forward or backward chaining) and search strategies (breadth or depth first).

Rule-Based Systems describe knowledge in terms of IF…THEN...ELSE. Decisions can be made using specific knowledge. They represent knowledge and decisions in ways that are understandable to human beings. Because of the rigid rule-base structure they can be poorer at handling uncertainty and imprecision. Typical rule-based systems have four fundamental components:

- the rules;
- an inference engine (or a semantic reasoner), that surmises information or acts depending on the interaction between the rules and the input(s);
- short-term memory;
- and user interfaces or alternative devices to input and output signals.

Case-Based Reasoning adapts solutions from earlier problems and applies them to existing problems. Solutions are stored in a database. The solutions can represent human experience. When a new problem is encountered, systems compare it with previous problems and select a problem that is most like the new problem. It then acts using the previous solution and records whether the action was successful or a failure. Case-Based Reasoning is effective at representative knowledge in a way that is easy and well-defined for humans, but they can also learn from previous examples by creating extra new solutions.

Case-based reasoning has been formalized as a process with four steps:

i. Retrieve: Recover cases from short-term memory that are applicable to solving a target problem. Cases include a problem, its solution, and, often, comments concerning the way that a solution was originated.
ii. Reuse: Map a solution from a previous case onto the target problem. The solution may need to be adapted automatically to fit a new situation.
iii. Revise: After mapping a previous solution onto a target situation, test the solution and revise it if necessary.
iv. Retain: Once successfully adapted then store the resultant occurrence as a new case within short term memory.

CBR is frequently described as an expansion of Rule-Based Systems. Both CBR and Rule-Based Systems are useful for denoting knowledge clearly but CBR systems can also learn from the past by automatically creating new cases.

A lot of expert systems are created using ‘shells'; ready-made programs that are expert systems (including inferencing and knowledge storage but lacking domain knowledge). Sophisticated expert systems can be created using ‘development environments'. Development environments are more flexible than shells. They provide ways for operators to employ their own inferencing and ways of representing knowledge.

Expert systems are probably the most mature methods from amongst the five tools considered here and lots of development tools and commercial shells are available. The building of a system can be relatively simple once domain knowledge has been extracted. Because they are relatively easy to develop, many applications have been created, for example for automatic robot programming and sequence planning.

B. Fuzzy logic

A rule-based expert system cannot handle a situation not explicitly included within their knowledge base (that is, situations not fitting within the ‘IF' statements within the rules). Rule-based systems cannot generate solutions when the encounter an unusual situation. They are consequently considered to be shallow systems which can fail in a ‘brittle' fashion, rather than gradually, as a human expert would.

Fuzzy logic reflects the qualitative and inexact nature of human reasoning. They can help an expert system to be more robust. Exact values for variables are exchanged for linguistic descriptions, represented by fuzzy sets. Based on this representation, the inferencing takes place. For example, an assembly speed of 35 thingamabobs per minute could be replaced by ‘normal' as a linguistic description of the variable ‘assembly speed. A fuzzy set defining the term ‘normal assembly speed ' might be:

normal assembly speed = 0.0/below 15 thingamabobs per minute +0.5/15−25 thingamabobs per minute +1.0/25−35 thingamabobs per minute +0.5/35−45 thingamabobs per minute +0.0/above 45 thingamabobs per minute.

The values 0.0, 0.5 and 1.0 are the degrees or grades of membership of the production ranges below 15 thingamabobs per minute (or above 45 thingamabobs per minute.), 15−25 thingamabobs per minute (35−45 thingamabobs per minute), and 25−35 thingamabobs per minute to the given fuzzy set. A grade of membership equal to 1 indicates full membership and a null grade of membership corresponds to total non-membership.

Knowledge within expert systems using fuzzy logic can be expressed as qualitative statements, (or fuzzy rules), such as ‘If apartment is at normal temperature, then set warmness inputs to normal'.

Reasoning procedures known as compositional rules of inference enable conclusions to be drawn by generalisation (interpolation or extrapolation) from qualitative information within a knowledge base. For example, when the normal assembly speed is perceived as ‘slightly below normal', a controlling fuzzy expert system may well determine that inputs should be increased to ‘slightly above normal'. Even though that conclusion may not have been covered by any fuzzy rule within the system.

Fuzzy Expert Systems use fuzzy logic to manage uncertainty produced by inadequate or partly corrupted data. Fuzzy logic uses a mathematical theory of fuzzy sets to mimic human logic. Humans easily deal with ambiguity when making decisions but computers still find it challenging.

Fuzzy logic has been used in mobile robotics, especially for control when domain knowledge has been imprecise. Fuzzy Logic is useful when there is imprecision. For instance, for object recognition and scene interpretation. Fuzzy expert systems are suitable for ambiguous and imprecise situations. They cannot learn because system values cannot be changed.

C. **Automatic knowledge acquisition**

Learning programs often need a set of examples to use but it can be time consuming and difficult to get domain knowledge into a knowledge base. That can create a bottleneck during the construction of an expert system. Automatic knowledge acquisition techniques were created to deal with that.

An example of an approach is ‘divide-and-conquer'. Here attributes are selected according to a strategy that divides an example set into several subsets. A decision tree is then built to classify examples. The decision tree represents knowledge that is generalised from a set of specific examples. This can then be used to handle situations not covered by the example set.

Another example is a ‘covering approach'. An inductive learning program endeavours to locate groupings of attributes that are uniquely shared by examples within classes and then form rules with the IF part as combinations of those attributes and the THEN part as the classes.

Another example is the use of logic programming in place of propositional logic to depict examples and characterise new concepts. That uses a more potent predicate logic to characterise training examples and background knowledge and to convey new concepts. That allows results from induction to be defined as unspecific first-order clauses with variables.

There are many learning programs such as:

- ID3 (a divide-and-conquer program),
- FOIL (an ILP system adopting generalisation/specialisation methods),
- AQ program (which follows a covering approach),
- and GOLEM, (ILP system based on inverse resolution).

Most of these sorts of programs generate crisp decision rules but some algorithms have been created that also produce fuzzy rules.

Automatic learning has been tricky to use with tele-operated mobile robots because they require a set of examples in a rigid format and few mobile robot problems are described easily within rigid sets of examples. Automatic learning is generally more suitable for problems with discrete or symbolic attribute values rather than those with continuous-values. A recent application of inductive learning is in the control of a laser cutting robot.

D. **Neural networks**

Neural networks can capture domain knowledge from examples. However, they do not archive the acquired knowledge in an explicit form such as in rules or decision trees. They can readily handle both discrete and continuous data. They also have a generalisation capability (as for fuzzy systems).

Neural network models distribute computation between several simpler units called neurons. Neurons are interconnected and operate in parallel so that, neural networks can be called parallel-distributed-processing systems.

The most popular neural network is the multi-layer perceptron, which is a feedforward network: all signals flow in a single direction from the input to the output of the network. Feedforward networks can perform static mapping between an input space and an output space: the output at a given instant is a function only of the input at that instant. Recurrent networks, where the outputs of some neurons are fed back to the same neurons or to neurons in layers before them, are said to have a dynamic memory: the output of such networks at a given instant reflects the current input as well as previous inputs and outputs.

Implicit ‘knowledge' is built into a neural network during training. Some networks can be trained by presenting them with typical input patterns and the corresponding expected output patterns. Errors between the actual and expected outputs are used to modify weights on connections between neurons. This is “supervised training”. In a multi-layer perceptron, the back-propagation algorithm for supervised training is often adopted to propagate the error from the output neurons and compute the weight modifications for the neurons in the hidden layers.

Some neural networks are trained in an unsupervised mode, where only the input patterns are provided during training and the networks learn automatically to cluster them in groups with similar features.

Artificial Neural Networks typically have inputs and outputs, with processing within hidden layers in between. Inputs are independent variables and outputs are dependent. ANNs are flexible mathematical functions with configurable internal parameters. To accurately represent complicated relationships, these parameters are adjusted through a learning algorithm. Once trained then ANNs can accept new inputs and attempt to predict accurate outputs. To produce an output, the network simply performs function evaluation. The only assumption is that there exists some continuous functional relationship between input and output data. Like expert systems, they have found a wide spectrum of applications in almost all areas of robotics, addressing problems ranging from modelling, prediction, control, pattern recognition and optimisation.

E. Genetic algorithms

A genetic algorithm is a stochastic optimisation procedure inspired by natural evolution. A genetic algorithm can yield a global optimum solution within a complex multi-modal search space without specific knowledge about a problem.

Potential solutions to a problem must be represented as strings of numbers known as chromosomes and there must be a means of determining the goodness of each chromosome. A genetic algorithm operates on a group or population of chromosomes at a time, iteratively applying genetically based operators such as cross-over and mutation to produce fitter populations containing better solution chromosomes. The algorithm normally starts by creating an initial population of chromosomes using a random number generator. It then evaluates each chromosome. The goodness values of the chromosomes are used in the selection of chromosomes for subsequent operations. After the cross-over and mutation operations, a new population is obtained and the cycle is repeated with the evaluation of that population.

Genetic algorithms have found applications in tele-operation problems involving complex combinatorial or multi-parameter optimisation. Some recent examples of those applications are in Robot Path Planning.

F. Combining systems

The purpose of a hybrid system is to combine the desirable elements of different AI techniques within a single system. The different AI methods each have their own strengths and weaknesses. Some effort has been made in combining different methods to produce hybrid techniques with more strengths and fewer weaknesses. An example is a Neuro-Fuzzy system which seeks to combine the uncertainty handling of Fuzzy Systems with the learning strength of Artificial Neural Networks.

The nodes of a Fuzzy Network are fuzzy rule bases and the connections between nodes are interactions in the form of outputs from nodes that are fed as inputs to the same or other nodes. A fuzzy network is a hybrid tool combining fuzzy systems and neural networks due to its underlying grid structure with horizontal levels and vertical layers. This tool can be suitable for modelling the environment because separate areas can be described as modular fuzzy rule bases interacting in sequential / parallel fashion and feed forward / feedback context. The main advantages from the application of this hybrid modelling tool are better accuracy due to the single fuzzification-inference-defuzzification and higher transparency due to the modular approach used. These advantages can be crucial because of uncertainties in the data and the interconnected structure of the environment.

3. Selection of the rule based expert system

A knowledge-based expert system was selected to assist in teleoperation using an inference mechanism because they are relatively simple and good at representing knowledge and decisions in a way that is understandable to humans.

A knowledge base was created that contained domain knowledge as a combination of 'IF–THEN' rules. An inference mechanism manipulated the knowledge to produce solutions to driving problems.

The rule-based system used IF…THEN...ELSE to make decisions. It had four basic components: a list of rules, an inference engine, temporary working memory and a joystick user interface.

4. Charting the environment in front of the robot

The sensors that were used were similar to those described in [27] and [28]. Ultrasonic sensors were mounted above the driving wheels on the front of the mobile robot. The time taken for an ultrasonic pulse to be reflected back to a sensor represented the distance to an obstacle. The robot is presented and explained in [29].

An imaginary potential field was placed around detected objects within the software [5][21]. As the ranges to obstacles altered then the sensor system modified pulse lengths. The range-finder progressively elongated pulses if obstacles were not being sensed so as to build-up the range until an obstacle was eventually detected. That technique provided an earlier warning of upcoming problems.

Histogramic In-Motion Mapping was used to filter out false readings. Volumes in front of the mobile robot were separated into left and right matrixes, with NEARBY, INTERMEDIARY and DISTANT compartments in each matrix. There was also a matrix that represented the volume in the center where the ultrasonic volumes intersected. That matrix represented the case when obstacles are detected by both sets of transducers. If an obstacle was perceived in front of the mobile robot then it was categorised as NEARBY, INTERMEDIARY or DISTANT. Transducers were mounted on the chassis of the mobile robot in such a way that the ultrasonic envelopes intersected and covered the environment in front.

If something was sensed then a quantity correlated with a cell was increased by a comparatively big amount, e.g.: ten to fifteen. Remaining cells reduced in value by a lesser amount, e.g.: five, downwards towards 0. The result was a histogrammic representation of obstacles in front of the robot. A cell quickly increased in value if an obstacle entered it. Random misreads simply incremented for a solitary misread before the cell then reduced to zero again. When obstacles appeared in other cells then those cells rapidly increased in value. If the obstacle moved away from the initial cell, then the value of the initial cell decreased back to 0. A consistent and dependable range was obtained within 0.4 seconds.

5. The user input

A Penny & Giles joystick was used that contained two potentiometers. Two A/D converters determined joystick location.

Data from the joystick was represented in Cartesian coordinates but were converted to polar coordinates.

$$|J| \angle\theta.$$

Where $|J|$ is a representation of the distance that the joystick has been shifted away from its central position. $|J|\angle\theta$ represented the velocity that an operative desired. $\angle\theta$ was the angle that represented the desired bearing from the robot position.

The confidence of the operative in their decision was assumed to be represented by how long the joystick remained in that position.

$$|J| = \sqrt{((JA*JA)+(JB*JB))} \qquad (1)$$

Where JA and JB represented the position of the joystick in Cartesian coordinates.

$|J|$ and θ established joystick position and from that, the chosen velocity (bearing and speed) could be calculated. The position and the level of confidence were logged within a matrix so that each cell within the matrix consisted of two values:

- "*Confidence*" specified the amount of time that a joystick had remained still.
- "*Magnitude*" quantified the desired speed.

Jstickin was used as an input to the rule base and gave a level of confidence about the intentions of the operatives.

Histogrammic representation carried out pseudo-integration. If operatives held their joysticks still, then the value of cells associated with that position grew. Other cells reduced in value. The cell that had the largest amount within it signified the position of the joystick.

The function *JstickArray* identified the position occupied by the joystick and *AngConf* increased for that position and the value of the empty cells reduced. Cell values reduced quickly but incremented more slowly.

JstickArray cells rose to their highest value in roughly 0.4s but decreased to 0 in roughly 100 to 200 milliseconds.

A weight was set to direct the rate of increasing in value and a separate weight was set to direct the rate of decreasing in value. These weights were determined through experimentation and testing. The two weights can be established and adjusted depending on the tasking and the abilities of the human operative.

6. Mobile robot kinematics

Kinematics are represented in Fig. 1 and are explained here. There were a pair of larger front wheels and a pair of smaller trailing casters at the back. Turning the driving wheels moved the robot. Turning them separately turned the robot so that the direction changed. If r represents the radius of the driving wheels, then 2r represents the diameter (Fig 2). Exploiting the notation used within [1], then W represents the distance between the driving wheels. The centre of gravity of the machine is C and P is placed at the junction of a line drawn through the centre of the machine and a line through the wheel axis. Distance between P and C is d.

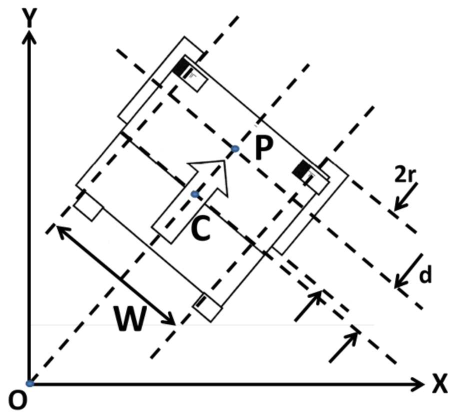

Fig. 2. Mobile robot geometry

Figure 3 shows the kinematics of the machine.

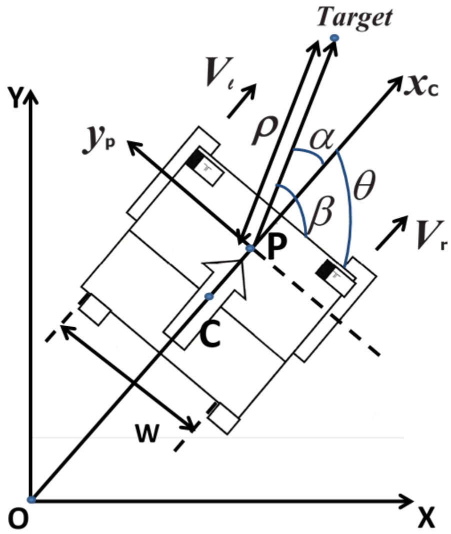

Fig. 3. Mobile robot kinematics

An assumption was made that there was not any slip between the floor and the wheels.

$$Vel_{tot} = 0.5 * (vel_{right} + vel_{left}) \tag{2}$$

$$\omega = 1/W * (vel_{right} - vel_{left}) \tag{3}$$

$$vel_{right} = r\omega_{right} \quad \text{and} \quad vel_{left} = r\omega_{left} \tag{4}$$

where v was the linear velocity ω was the angular velocity. Position in global coordinates was [O X Y]. In vector notation that was:

$$\boldsymbol{q} = [\ x\ y\ \theta]^T \tag{5}$$

where x and y were the global coordinates of P as shown in Fig. 2. θ was the orientation of $[\ P\ x\ y\ ]$. They were the local coordinates as shown in Fig. 3., from the horizontal axis. These defined configuration (5). The machine was assumed to be rigid. Wheels were assumed not to slip. That meant that the machine could only move in a direction that was normal to the axis of the wheels. That meant that the velocity where the wheel contacted the floor and was orthogonal to the wheel plane was 0.

$$(dy/dt) \cos\theta - (dx/dt) \sin\theta - d\theta/dt = \text{zero} \tag{6}$$

Kinematics restrictions are not time dependent, so they can be considered as

$$A^T (\boldsymbol{q})\ d\boldsymbol{q}/dt = \text{zero} \tag{7}$$

where $A(\boldsymbol{q})$ represents the input matrix that is associated with the constraints. Then

$$C^T A(\boldsymbol{q}) = 0 \tag{8}$$

where $C(\boldsymbol{q})$ is a full-rank matrix of the set of linearly independent vector fields covering the null space of $A^T(\boldsymbol{q})$. v_{tot} is a function of vector time found from equations (7) and (8) for time t.

$$d\boldsymbol{q}/dt = C(\boldsymbol{q})\ v_{tot} \tag{9}$$

For the machine, the constraint matrix shown in (6) becomes

$$\mathbf{A}^T (\boldsymbol{q}) = [-\sin\theta\ \cos\theta\ -d] \tag{10}$$

and

$$\boldsymbol{v}_{tot} = [v \ \omega \ ]^T \quad (11)$$

Where v is the linear velocity and ω is the angular velocity of P (along the machine axis). So, kinematics (9) can be represented in a dq/dt matrix. Because the machine is considered to only move forwards then $v = - v_{tot}$ and the system can be simplified and represented by a simpler matrix. Steering angle and wheel velocity were generated by a controller such that:

SAngle = ($v_{left} - v_{right}$) / W,

SAngle was used to drive the machine to follow a desired path.

7. Control and rules

The controller calculated ω and v to move the machine from a configuration, e.g.: $\rho_0 \, \alpha_0 \, \beta_0$, to a new position and orientation. Considering linear control[30]

$$v = K_\rho \rho \quad (12)$$

$$\omega = K_\alpha \alpha + K_\beta \beta \quad (13)$$

A matrix can depict this closed-loop system to drive the machine to

$$(\rho, \alpha, \beta) = (0,0,0).$$

Where (0,0,0) represented a target destination.

Control was successfully simulated and then the controller was tested on the robot. Joystick inputs and the output of the ultrasonic sensors were merged using a rule set that aimed to avoid objects. The first set of rules that were tested were a combination of 4 inputs (fig 4). They were: 1.Joystick steering angle; 2. Range to obstacles that both sensors detected; 3.Range to obstacles on the left and 4. range to obstacles on the right.

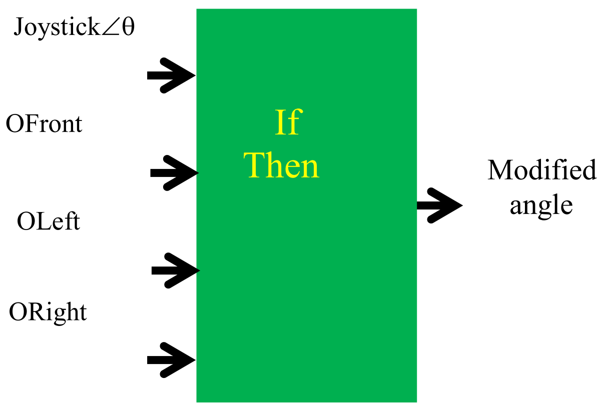

Fig. 4. The first set of rules tested.

The steering angle used by the controller was modified by the input from the ultrasonic sensors. The input form the ultrasonic sensors represented the environment ahead. The resultant movement efficient and safe. If $\angle\theta$ was to the right then the robot tended to rotate clockwise. If $\angle\theta$ was to the left, then it turned anticlockwise.

The first system performed satisfactorily if the operatives could see their mobile robots but not if the operatives could not see the them. To overcome this limitation, the systems were enhanced and upgraded. The initial set of rules were revised to include a target destination. That target destination could assist the operatives if they could not see what was happening (fig. 5.). The new revised system now included a target point as well as the original environmental information (about the volume ahead) and the steering angle provided by the joystick. That addition considerably enlarged the number of rules.

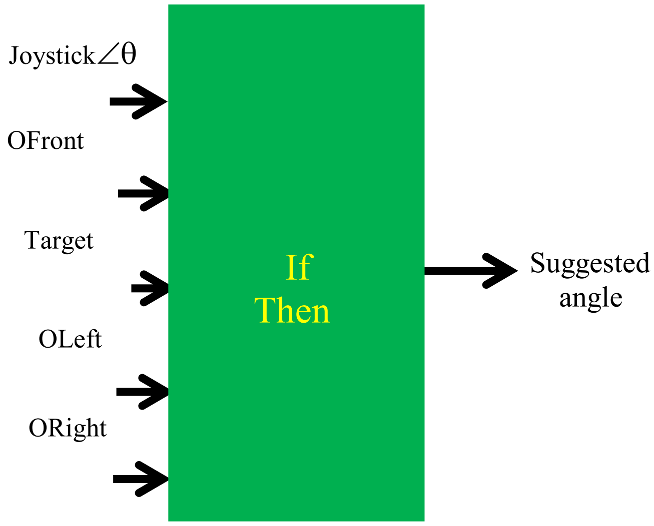

Fig. 5. The augmented system with a Target added.

Some of the rules are shown here in their updated form:

CASE 1 - the destination and an obstacle are to the left:

Rule 1: *If* Jstick = 0^o *and* OLeft=INTERMEDIARY *and* ORight $\leq$ DISTANT *and* OFront $\leq$ DISTANT *and* Target Angle = 70^o, *then* recommended adjustment to the steering angle = 0^o

Rule 2: *If* Jstick = 0^o *and* OLeft=INTERMEDIARY *and* ORight $\leq$ DISTANT *and* OFront $\leq$ DISTANT *and* Target Angle = 60^o, *then* recommended adjustment to the steering angle = -10^o

Rule 3: *If* Jstick = 0^o *and* OLeft=INTERMEDIARY *and* ORight $\leq$ DISTANT *and* OFront $\leq$ DISTANT *and* Target Angle = 50^o, *then* recommended adjustment to the steering angle = -25^o

CASE 2 - destination is to the right and an obstacle is to the right:

Rule 4: *If* Jstick = 0^{o} *and* OLeft ≤ DISTANT *and* ORight = INTERMEDIARY *and* OFront ≤ DISTANT *and* Target Angle=75^{o}*, then* recommended adjustment to the steering angle = 15^{o}

Rule 5: If Jstick=0^{o} *and* OLeft= ≤ DISTANT and ORight = INTERMEDIARY *and* OFront ≤ DISTANT *and* Target Angle = 60^{o}, *then* recommended adjustment to the steering angle = 30^{o}

Rule 6: *If* Jstick=0^{o} *and* OLeft= ≤ DISTANT *and* ORight = INTERMEDIARY *and* OFront ≤ DISTANT *and* Target Angle = 30^{o}, *then* recommended adjustment to the steering angle = 25^{o}

CASE 3 - destination is to the right and an obstacle is ahead:

Rule 7: *If* Jstick = 0^{o} *and* OLeft= NEARBY *and* ORight = NEARBY *and* OFront ≤ DISTANT *and* Target Angle =20^{o}, *then* recommended adjustment to the steering angle = 15^{o}

Rule 8: *If* Jstick = 0^{o} *and* OLeft= NEARBY *and* ORight = NEARBY *and* OFront ≤ DISTANT *and* Target Angle =25^{o}, *then* recommended adjustment to the steering angle = 20^{o}

Rule 9: *If* Jstick= 0^{o} *and* OLeft= NEARBY *and* ORight = NEARBY *and* OFront ≤ DISTANT *and* Target Angle = 30^{0}, *then* recommended adjustment to the steering angle = 25^{o}

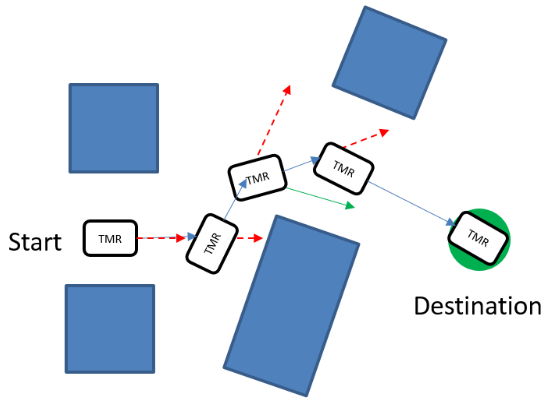

Fig. 6. Mobile robot being driven through objects (blue boxes) using the updated set of rules with calculated directions (red dashed line) and approach directions (blue solid line).

The new system worked as satisfactorily as the previous system with the new rule set but it worked especially well when the operatives could not see the mobile robot.

The mobile robot path is in Fig. 6. with the additional rules being applied. The red dashed lines and arrow heads are the angles to the destination. The objects around the robot are shown by the blue boxes and the approach directions are shown by blue solid lines.

8. Experimentation and results

A typical simulation of the system is shown in Fig. 7.

After the algorithms had successfully been tested in simulation, then the hardware and software were mounted onto the mobile robot base. Standard test routes at Portsmouth University were used for the tests.

Obstacles were avoided. When and object was detected by the sensors that was relatively near to the robot, then the mobile robot steered away from any potential impact. The operator could overrule the system by moving their joystick if, for example, an operative wanted to move the mobile robot closer to something.

The system began to take effect when sensors detected an object as DISTANT or nearer. If the ultrasonic transducers sensed something in front of the robot while it was moving towards a destination, then the robot rotated to pass along the side of the obstacle. When nothing was ahead, and the joystick was held forward, then the mobile robot headed in the direction of the destination. That reduced time taken by a significant amount when the operator could not see the mobile robot. The modified rules quickly changed the bearing and course so that the mobile robot moved in the direction of the destination.

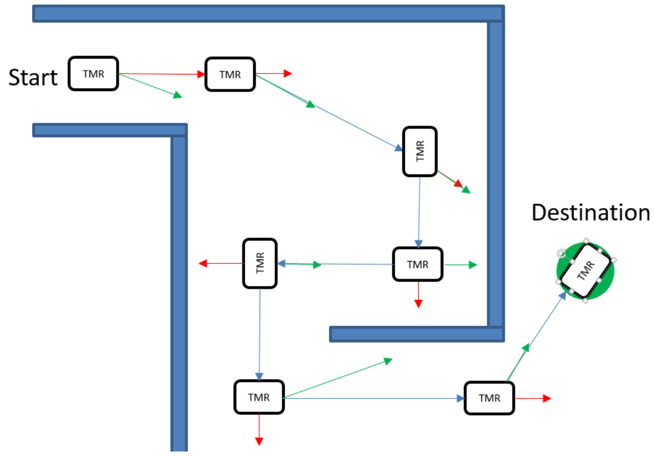

Fig. 7. Typical simulation of a robot path with the system using the revised rules and exhibiting the behaviour of the robot as it avoids local minima (for example the corners of the inner walls).

Results from a real time experiment and from a simulation of the mobile robot are presented in Fig. 8. and Fig. 6. To show the way in which the system was validated.

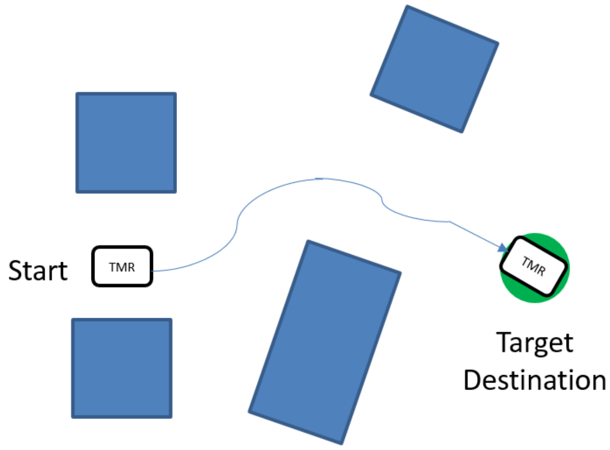

Fig. 8. Results from applying the revised rules in a real-time experiment.

The performance of the system was compared with the rests form using the system described in [1]. This system tended to complete tasks faster than the earlier system. Figure 9 compares the time taken by the two systems. In each case, the robot was steered along a standard route.

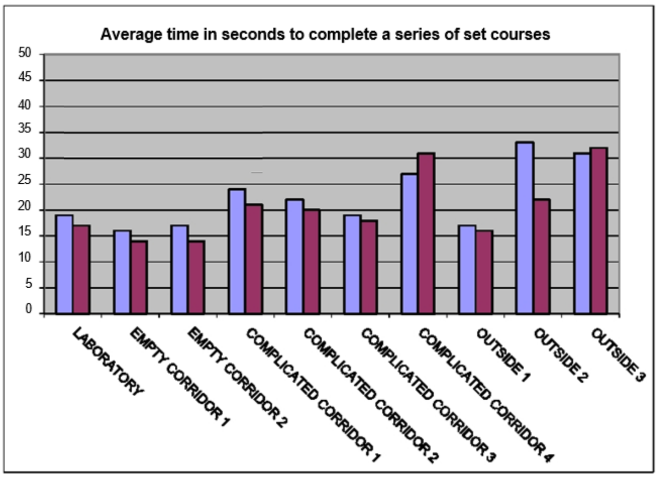

Fig. 9. Comparing the new system against a mobile robot without any sensors to assist the operator. Average time taken to complete a series of set courses is shown. Right hand bars show the time taken when the ultrasonic sensors are assisting the operator and left hand bars show the time taken when the sensor system is switched off.

The new systems completed courses more quickly in most cases. Figure 10 shows two anomalies. The more complicated routes required the mobile robot to turn more often. The simpler routes did not need a sensor system as they could easily be achieved (and more quickly) by a tele-operator without any sensors to assist them.

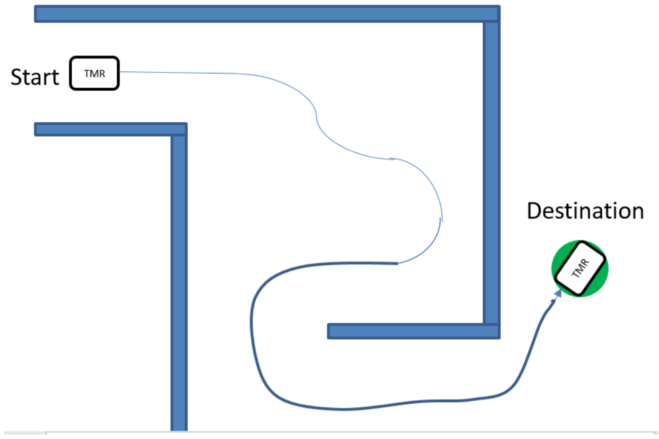

Fig. 10. Mobile robot path when the updated rule set was applied and the tele-operative can't see the mobile robot.

Including destination as an additional input to the rule base made driving less effective in some of the easier sections of routes when the operative could see the robot and the surroundings. The tele-operative didn't need sensors to assist them. Two routes are shown as examples in Fig. 11 and Fig. 12.

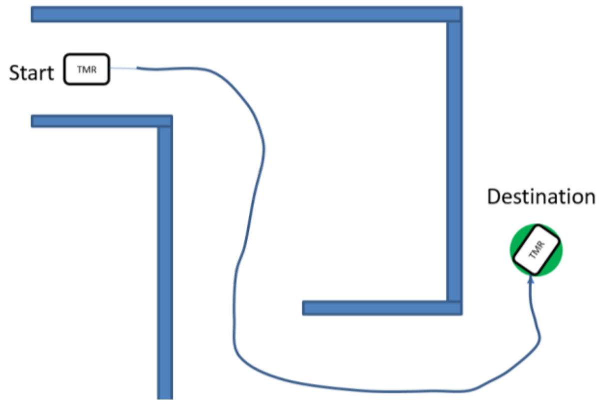

Fig. 11. Mobile robot path when the updated rule set was applied and the tele-operative can see the mobile robot.

The rules tended to attract the mobile robot toward the destination, as shown in Fig. 10. The first rule set was only influenced in direction by the joystick input. Although the path taken was usually less efficient when the tele-operator could not see the mobile robot, the route was always completed. The difference in the routes taken is presented in Fig. 12.

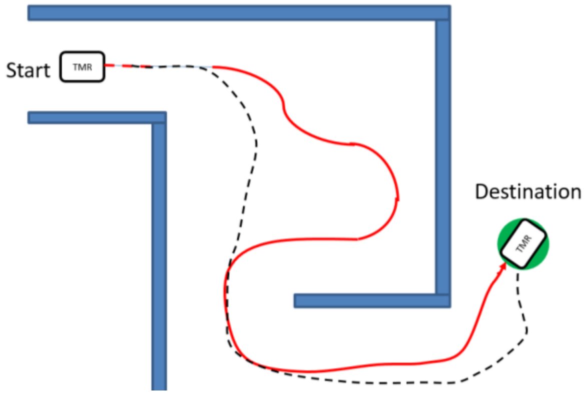

Fig. 12. The different paths taken by a mobile robot when using the revised rules. The dotted blue line shows the route taken when the tele-operator can see the robot and the solid red line shows the route taken when the tele-operator cannot see the mobile robot and the robot must rely only on the feedback from the ultrasonic sensors.

When an experienced tele-operator can see the mobile robot, and can drive well then the operative can override the rule set and take a more efficient route.

The robots efficiently reached destinations.

The rule based methods gave a quicker response in most cases and reduced computation time compared with other recent approaches. Figure 11 shows a real-time path.

The robot successfully avoided both moving and static objects. When sensors detected close objects, then the robot turned to avoid collision.

Avoiding collisions was a high priority and that would primarily override other behaviour but if the operator held the joystick in the same position (roughly) then that input from the joystick would be integrated over time and the wishes if the operative would override the system.

When an obstacle was detected ahead while the robot was moving toward a destination then the robot exhibited wall-following behaviour; the robot rotated to align with an object and then moved parallel to the side of the object.

If the sensors were not detecting any objects, then the robot moved in a direction that was an average of the angle directed by the joystick position and the angle to the destination. If the joystick requested a direction that was aligned with the direction to the destination, then the robot moved towards the destination.

The rule-based system described here performed well when results were compared with those from other recent systems.

9. Discussion

Artificial intelligence has produced several powerful tools. This paper has reviewed some of those tools: knowledge-based systems, fuzzy logic, automatic learning, neural networks, ambient intelligence and genetic algorithms

The rule-base system selected was less good at handling uncertainty and is poor at handling imprecision because of the rigid structure. Case-Based Reasoning systems are often considered to be an extension of Rule-Based Systems. They are good at representing knowledge in a way that is clear to humans, but they also have the ability to learn from past examples by generating additional new cases.

Case-Based Reasoning could have been used because that can adapt solutions from previous problems to current problems. Solutions could be stored within a database. When a problem occurred that a system had not experienced, it could compare with previous cases and select one that was closest to the current problem. It could then update the database depending upon the outcome.

Without statistically relevant data for backing and implicit generalization, there is no guarantee that any generalization would be correct. However, all inductive reasoning where data is scarce is inherently based on anecdotal evidence.

The use of AI brings us to a point in history when our human biology can appear too slow and over-complicated. To overcome this, we are beginning to mix sensor systems and some powerful new technologies to overcome those weaknesses, and the longer we use that technology, the more we are getting out of it. We use less energy, space, and time, but get more and more assembly output for less cost. The AI exceeded human performance in several tasks. As computers merge with us more intimately and we combine our brain power with computer capacity, then teleoperation should become easier and more efficient. AI can reduce mistakes and increase efficiency. Time taken therefore reduces.

10.Conclusions

Applications of the AI tools discussed in this paper have become more widespread due to the power and affordability of present-day computers. Many new mobile robot applications may emerge and greater use may be made of hybrid tools that combine the strengths of two or more of the tools reviewed here. The tools have minimal computation complexity and can be implemented on single robots or systems with low-capability microcontrollers.

The rule-based systems were robust and safe. They were simple and efficient in helping with driving. The rule based techniques were employed effectively. The robot quickly detected obstacles ahead and assisted operatives with their tasks.

Laboratory testing was compared with simulated paths and the rules were validated. Systems compared favourably with other contemporary structures described in the literature and that also validated the methods and systems.

Ongoing research is exploring the integration and combining of diverse AI techniques to extract the best from each technique.

References

[1] D.R. Parhi, and M.K. Singh “”Rule-based hybrid neural network for navigation of a mobile robot,” Proc. IMechE Part B: J. Engineering Manufacture, Vol. 224, 2009, pp 11103-1117.

[2] A.V. Nguyen, L.B. Nguyen, S. Su, and H.T. Nguyen “Shared Control Strategies for Human - Machine Interface in an Intelligent Robot,” 35th Annual Int Conf of IEEE-Engineering-in-Medicine-and-Biology-Society (EMBC), Osaka, JAPAN. Book Series: IEEE Engineering in Medicine and Biology Society Conference Proceedings, 2013, pp: 3638-3641.

[3] Z.R. Parhi, S.K. Pradhan, A.K. Panda, and R.K. Behra, “The stable and precise motion control for multiple mobile robots,” Appl. Soft Comput vol 9(2), 2009, 477- 487.

[4] D. A. Sanders. D. Ndzi, S. Chester and M. Malik “Adjustment of tele-operator learning when provided with different levels of sensor support while driving mobile robots”. Proc’ SAI Intelligent Systems Conference 2016, London, UK . *In press*.

[5] K. T. Song, and C. C. Chen, “Application of asymmetric mapping for mobile robot navigation using ultrasonic sensors,” J. Intell. Robotic Syst., 1996, 17(3), 243-264.

[6] R. Huq, G. K. I. Mann and R. G. Gosine, “Mobile robot navigation using motor schema and fuzzy context dependent behaviour modulation,” Appl. Soft Comput., 2008, 8(1), 422-436.

[7] M. Begum, G. K. I. Mann, and R. G. Gosine, “Integrated fuzzy logic and genetic algorithmic approach for simultaneous localization and mapping of mobile robots,” Appl. Soft Comput., 2008, 8(1), 150-165.

[8] M. Bennewitz and W. Burgard, “A probabilistic method for planning collision-free trajectories of multiple mobile robots,” In Proc 14th Euro Conf on AI, Berlin, Germany, 20-25 August 2000, pp. 9-15 (ECAI).

[9] W. Gueaieb and M. S. Miah, “An intelligent mobile robot navigation technique using RFID technology,” IEEE Trans. Instrum. Measmt, 2008, 57(9), 1908-1917.

[10] C. L. Hwang and N. W. Chang, “Fuzzy decentralized sliding-mode control of a car-like mobile robot in distributed sensor-network spaces,” IEEE Trans. Fuzzy Syst., 2008, 16(1), 97-109.

[11] A.V. Nguyen, and L.B. Nguyen, S. Su, H.T. Nguyen, “The Advancement of an Obstacle Avoidance Bayesian Neural Network for an Intelligent Robot”, 35th Int Conf of IEEE-Engineering-in-Medicine-and-Biology-Society (EMBC), Osaka, JAPAN. Book Series: IEEE Eng in Medicine and Biology Society Conf Proc, 2013, pp: 3642-3645.

[12] D. A. Sanders. “Using a self-reliance factor for a disabled driver to decide on the share of combined-control between a powered wheelchair and an ultrasonic sensor system”. IEEE Trans on Neural Systems and Rehabilitation Engineering. *In press*.

[13] D. Sanders, I. Stott, J. Graham-Jones, A. Gegov, and G.E. Tewkesbury, “Expert system to interpret hand tremor and provide joystick position signals for tele-operated mobile robots with ultrasonic sensor systems”. Ind Rob 38 (6). 2011, pp. 585-598.

[14] D.C. Robinson, D.A. Sanders, and E. Mazharsolook. “Ambient intelligence for optimal manufacturing and energy efficiency”. Assem. Autom. Vol 35 (3), 201, pp: 234-248

[15] D.A. Sanders, G. Tewkesbury, A. Gegov, *et al*. “Fast transformations to provide simple geometric models of moving objects”. Proc’ Intelligent Robotics & Application Confs, Pt I Vol 9244, 2015, pp: 604-615.

[16] D. Sanders, H. Sanders, D Ndzi and A. Gegov, “Rule-based system to assist a tele-operator with driving a mobile robot” IEEE Proc of SAI Intelligent Systems Conf, 2016, pp: 987–994

[17] J. Larsson M., Broxvall and A. Saffiotti, “Laser-based corridor detection for reactive Navigation. Ind Rob: An int' jnl, 35 (1), 2008, pp 69-79.

[18] D. Sanders, J. Graham-Jones, and A. Gegov, “Improving ability of tele- operators to complete progressively more difficult mobile robot paths using simple expert systems and ultrasonic sensors,” Ind Rob: An int' jnl, 37 (5). 2010, pp. 431-440.

[19] S. Lee “Use of infrared light reflecting landmarks for localization. Ind Rob: An int' jnl 36 (2), 2009, pp: 138-145.

[20] V. Milanes, J. Naranjo, C. Gonzalez *et al*, "Autonomous vehicle based in cooperative GPS and inertial systems", Robotica 26, 2008, pp 627-633.

[21] D. Sanders and I. Stott, "A new prototype intelligent mobility system to assist tele-operated mobile robot users", Ind Rob 26(6), 09, pp: 466-475.

[22] Y. C. Chang and Y. Yamamoto "On-line path planning strategy integrated with collision and dead-lock avoidance schemes for wheeled mobile robot in indoor environments," Ind Rob: An int' jnl, 35 (5), 2008, pp 421-434.

[23] D. Sanders, "Progress in machine intelligence", Ind Rob 35 (6), 2008, pp: 485-487.

[24] D. A. Sanders, G. Lambert, and L. Pevy, "Pre-locating corners in images in order to improve the extraction of Fourier descriptors and subsequent recognition of shipbuilding parts", Proc' Inst Mech Eng PART B-Jrnl of Engineering manufacture 223, vol 9, 2009, pp: 1217-1223.

[25] D. Sanders, "Comparing speed to complete progressively more difficult mobile robot paths between human tele-operators and humans with sensor-systems to assist," Assem. Autom. vol 29 (3), 2009, pp 230-248.

[26] D. Sanders, "Analysis of the effects of time delays on the tele-operation of a mobile robot in various modes of operation," Ind Rob vol 36 Issue: 6, 2009, pp: 570-584. 2.

[27] D. Sanders and A. Gegov. "Different levels of sensor support changes the learning behaviour of wheelchair drivers". Assistive Technology. *In press*.

[28] D. A. Sanders and N.Bausch, (Ed H. Liu). "Improving Steering of a Powered Wheelchair Using an Expert System to Interpret Hand Tremor". Proc of Intelligent Robotics and Applications (Icira 2015), Pt Ii Volume: 9245, pp: 460-471. 2015.

[29] D. Sanders, I. J. Stott and D. C. Robinsosn e*t al.* "Analysis of successes and failures with a tele-operated mobile robot in various modes of operation". Robotica 30, pp: 973-988. 2012.

[30] D. A. Sanders, G. E. Tewkesbury, and I. J. Stott, *et al*. "Simple expert systems to improve an ultrasonic sensor-system for a tele-operated mobile-robot". Sensor Review 31 (3), pp 246-260. 2011.

[31] D. Sanders and A. Geov. "AI tools for use in Assem. Autom. and some examples of recent applications. Assem. Autom. Vol 33 (2), pp: 184-194. 2013.

[32] D. A. Sanders, G. E. Tewkesbury, and D. Ndzi *et al*. "Improving automatic robotic welding in shipbuilding through the introduction of a corner-finding algorithm to help recognise shipbuilding parts". Journal of Marine Science and Technology Vol 17 (2), pp: 231-238. 2012.

[33] D. Sanders, G. Lambert, and J. Graham-Jones *et al*. "A robotic welding system using image processing techniques and a CAD model to provide information to a multi-intelligent decision module". Assem. Autom. Vol: 30 (4), pp: 323-332. 2010.

[34] D. A. Sanders, and G. E. Tewkesbury. "A pointer device for TFT display screens that determines position by detecting colours on the display using a colour sensor and an Artificial Neural Network". Displays Vol 30 (2), pp: 84-96. 2009.

[35] J. Bergasa-Suso, D. A. Sanders, and G. E. Tewkesbury. "Intelligent browser-based systems to assist Internet users". IEEE Transactions on Education Vol 48 (4), pp: 580-585. 2005.

[36] D. A. Sanders, Y. Tan, I Rogers *et al*. "An expert system for automatic design-for-assembly". Assem. Autom. 29 (4), pp: 378-388. 2009.

[37] D. A. Sanders, G. Lambert, and L. Pevy. "Pre-locating corners in images in order to improve the extraction of Fourier descriptors and subsequent recognition of shipbuilding parts". Proc of IMechE Part B-Journal of Engineering Manufacture 223 (9), pp: 1217-1223. 2009.

[38] D. Sanders. "Controlling the direction of walkie type forklifts and pallet jacks on sloping ground". Assem. Autom. 28 (4), pp: 317-324. 2008.

[39] D. A. Sanders. "Progress in machine intelligence". Ind Rob: An int' jnl 35 (6), pp: 485-487. 2008.

[40] A. Geov, N. Gobalakrishnan and D. A. Sanders. “Rule base compression in fuzzy systems by filtration of non-monotonic rules”. Journal of Intelligent & Fuzzy Systems Vol 27 (4), pp: 2029-2043. 2014.